UC SANTA BARBARA engineering

Solving large-scale games arising in cyber-awareness

João P. Hespanha   Shaunak Bopardikar

Joint work with: M. Prandini, A. Borri, M. D. Di Benedetto

UC SANTA BARBARA engineering

Outline

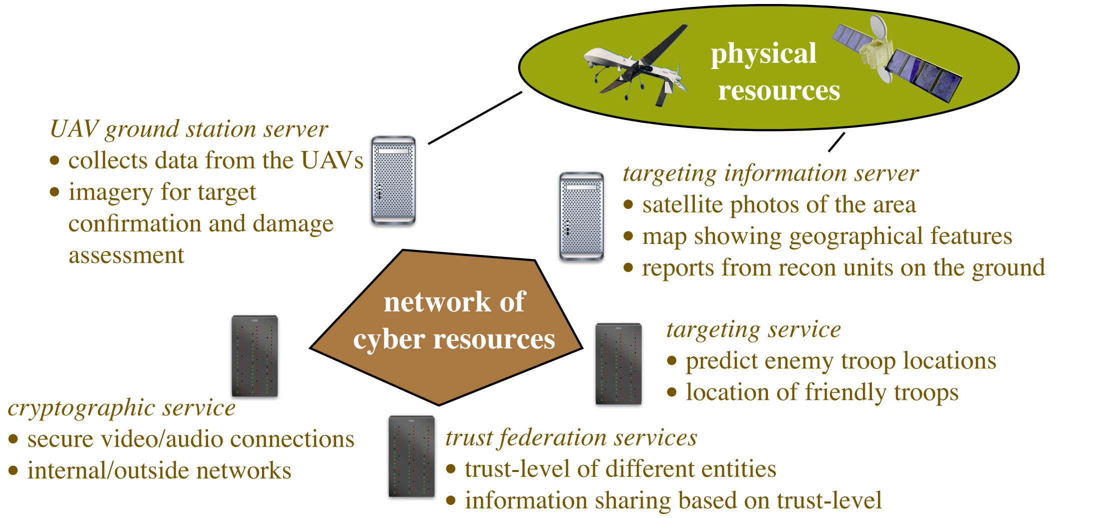
- Motivation (Cyber Security)
- Game Theory Background
- Sampled Saddle Points (SSP)
- Case Studies

Motivating Scenario

UC SANTA BARBARA engineering

Mission: Targeting, authorization to launch weapon, damage assessment, clean up

- Combination of forces, multiple countries (internal & external data networks)
- Mix of automated and human decisions based on real-time feedback
- Redundant systems, multiple levels of performance may acceptable
- System is vulnerable to attacks both at physical & cyber resources



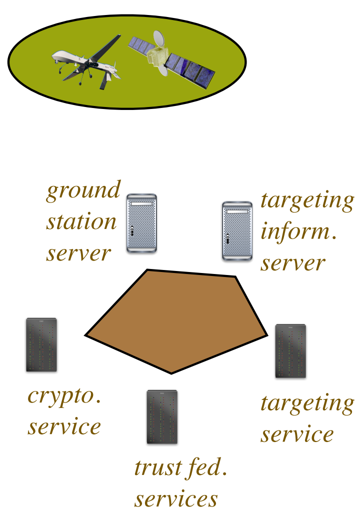
ARO MURI: Cyber Situation Awareness

Key ingredients

UC SANTA BARBARA engineering

Mission: Targeting, authorization to launch weapon, damage assessment, clean up

- Combination of forces, multiple countries (internal & external data networks)
- Mix of automated and human decisions based on real-time feedback
- Redundant systems, multiple levels of performance may acceptable
- System is vulnerable to attacks both at physical & cyber resources

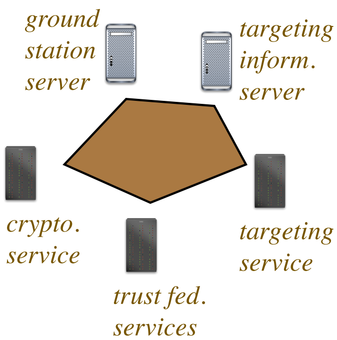


Key ingredients

1. *Mission* is characterized by
   - "spatial" pattern: which combination of resources are required for success?
   - "temporal" pattern: which resources are needed when?
2. *Actors*
   - blue & red teams
   - opposite goals
3. *Rules of engagement*
   - which actions are allowed?
   - what are the consequences of the actions?
4. *Information structure*
   - what information is available to take actions?
   - can one actor see the effect of the other's actions?

Cyber Awareness Games

UC SANTA BARBARA engineering



Mission:

1. target information server (TIS) gathers data
   UAV ground station server (GSS) gathers data
2. targeting service (TS) queries TIS and generates candidate target coordinates
3. human operator queries GSS for near-real-time imagery
4. human operator operator either
   accepts target or
   requests alternative coordinates (back to 2.)

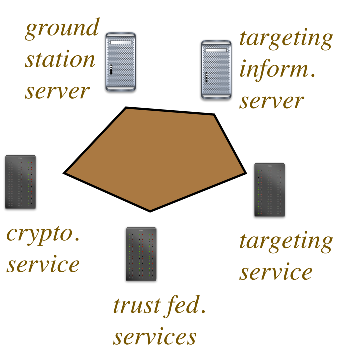
...

10. allied ground forces complete clean up
11. end-of-mission confirmation received by command center

- *at different points in time, different cyber-resources need to be available*
- *some level of redundancy allows for a mission to be completed using different configurations of resources*

Cyber Awareness Games

UC SANTA BARBARA engineering



Actors & their ***Cyber actions:***

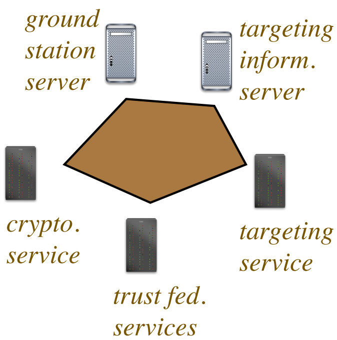
Blue forces may...

- rate-limit:
  constrain multiple connections from same IP
  constrain overall rate of service responses
- confine:
  enable/disable services or disallow new connections
  kill processes and/or de-authorize users
- re-instate:
  reboot host
  reinstall host OS to uncompromised state

Red forces may...

- compromise:
  gain access to hosts
  tamper with data
- disable:
  disable services
  disable hosts
- degrade performance:
  launch DOS attacks
  compromise routing or transport

Cyber Awareness Games



Actors & their ***Cyber actions:***

Blue forces may...

- rate-limit:
  - constrain multiple connections from same IP
  - constrain overall rate of service responses
- confine:
  - enable/disable services or disallow new connections
  - kill processes and/or de-authorize users
- re-instate:
  - reboot host
  - reinstall host OS to uncompromised state

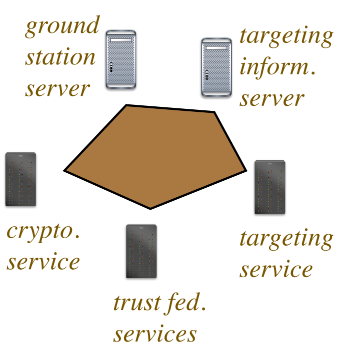
Re

- *Blue's trade-off: turning off all services will guarantee that red cannot compromise cyber infrastructure, but will also prevent mission completion*
- *Detailed knowledge of the possible red actions cannot be assumed a-priori (due to the potential for unknown vulnerabilities)*

Will provide estimates for mission success for different levels of unknown

ort

Cyber Awareness Games



Information structure:
both sides only have a partial view of mission's state

Blue has access to

- service current availability
- alerts form packet sniffing systems (detection of known malware)
- anomaly-based intrusion detection systems (deviations from normal behavior)
- OS and network logs

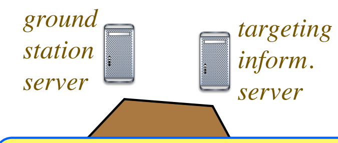
but difficult to conclusively determine if

- a host has been compromised
- a re-instate measure succeeded at "cleaning" a host

Red may also have difficulties in determining if

- gained access to a real host or to a "sand-box"
- succeed in preventing a key mission step
- a previously compromised host has been cleared

Cyber Awareness Games



Information structure:
both sides only have a partial view of mission's state

Blue has access to

***Partial information** is a crucial aspect of the problem*

- *Our estimates of the **current** "state" of the mission depend on what we believe the adversary might have done in the past*
  - *infinite belief recursion (I think, that she thinks, that I think, …)*
- *No known solutions based on **separation** between estimation & control (cannot independently estimate state & then decide best action)*
- *No known solutions using **dynamic programming** (thus high complexity and inability to prune)*

Red may also have difficulties in determining if

- gained access to a real host or to a "sand-box"
- succeed in preventing a key mission step
- a previously compromised host has been cleared

Matrix Game Abstraction

Two players:
P1 - defender (minimizer)
P2 - attacker (maximizer)

Each player selects a policy:
S1 - set of available policies for P1
S2 - set of available policies for P2

All possible game outcomes can be encoded in a matrix (2D-array), jointly indexed by the actions of the players

game outcome when
P1 selects action *i*
P2 selects action *j*

$$\begin{bmatrix} & \vdots & \\ \cdots & a_{ij} & \cdots \\ & \vdots & \end{bmatrix} \leftarrow \text{P1's policy}$$

$\uparrow$
P2's policy

Attention! To allow for dynamic partial information games, "policy" must be understood in a feedback sense:

What will be my response to each possible observation?

$$\text{policy} : \text{observations} \mapsto \text{actions}$$

Pure Security Policies

Security level for P1 (minimizer): $$\bar{V} := \underbrace{\min_i \underbrace{\max_j a_{ij}}_{\text{for each own action, consider worst adversary response}}}_{\text{pick best worst-case (called \textit{security policy})}}$$

guaranteed performance level against any adversary's choice

Security level for P2 (maximizer): $$\underline{V} := \max_j \min_i a_{ij}$$

Pure Security Policies

Security level for P1 (minimizer): $$\bar{V} := \underbrace{\min_i \underbrace{\max_j a_{ij}}_{\text{for each own action, consider worst adversary response}}}_{\text{pick best worst-case (called \textit{security policy})}}$$

guaranteed performance level against any adversary's choice

Security level for P2 (maximizer): $$\underline{V} := \max_j \min_i a_{ij}$$

But ...

(Pure) security levels/policies can be very conservative -- implicitly assume

1. other player knows our policy ahead of time and
2. selects its response based on that knowledge

think R.P.S.

Randomized Policies

UC SANTA BARBARA engineering

Mixed policies $\equiv$ selecting policies randomly according to a carefully chosen distributions (as opposed to always selecting fixed policy)

Mixed security level for P1 (minimizer): $\bar{V} := \min_y \max_z \mathrm{E}[a_{ij}]$

optimization over probability distribution y used to select policy i

optimization over probability distribution z used to select policy j

Mixed security level for P2 (minimizer): $\underline{V} := \max_z \min_y \mathrm{E}[a_{ij}]$

Randomized Policies

UC SANTA BARBARA engineering

Mixed policies $\equiv$ selecting policies randomly according to a carefully chosen distributions (as opposed to always selecting fixed policy)

Mixed security level for P1 (minimizer): $\bar{V} := \min_y \max_z \mathrm{E}[a_{ij}]$

optimization over probability distribution y used to select policy i

optimization over probability distribution z used to select policy j

Mixed security level for P2 (minimizer): $\underline{V} := \max_z \min_y \mathrm{E}[a_{ij}]$

- mixed security levels for both players always match (**minmax Theorem**)

$$V := \min_y \max_z \mathrm{E}[a_{ij}] = \max_z \min_y \mathrm{E}[a_{ij}]$$

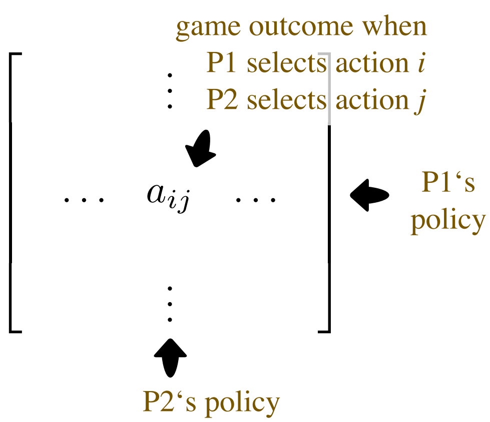
- non-conservative solutions — other player can “corner” us into the security level without knowing our policy

Matrix Game Abstraction

Two players:
- P1 - defender (minimizer)
- P2 - attacker (maximizer)

Each player selects a policy:
- S1 - set of available policies for P1
- S2 - set of available policies for P2

All possible game outcomes can be encoded in a matrix (2D-array), jointly indexed by the actions of the players



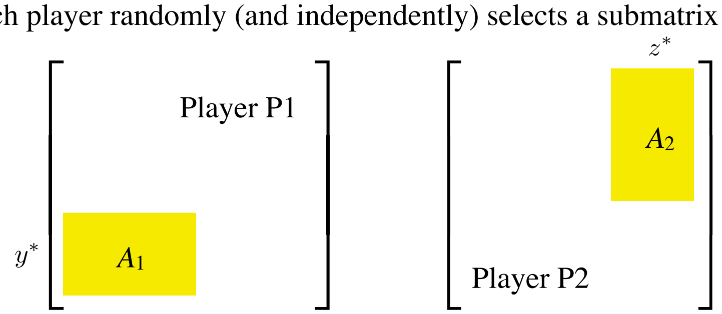
Attention! To allow for dynamic partial information games, "policy" must be understood in a feedback sense:

What will be my response to every possible observation?

- *huge # of possible choices*
- *for most interesting games, it is not feasible to even construct the whole matrix*

Sampled Saddle Point (SSP) Algor.

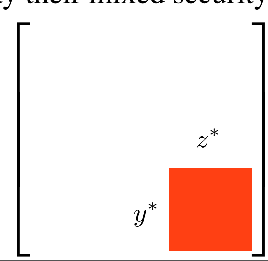
1. Each player randomly (and independently) selects a submatrix of the overall game



in very large games, submatrices will likely be non-overlapping

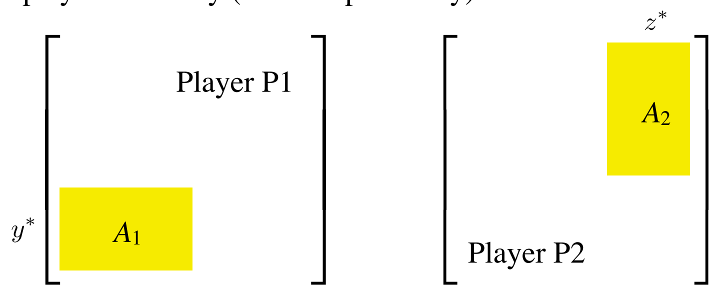
2. Each player solves its subgame (as if it were the whole game) and computes
   - mixed security levels: $V(A_1)$ & $V(A_2)$
   - corresponding security policies: y^*, z^*

3. Players play their mixed security policies against each other



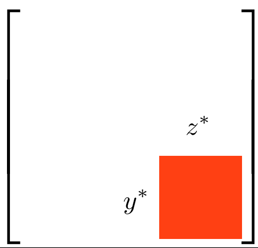
Sampled Saddle Point (SSP) Algor.

1. Each player randomly (and independently) selects a submatrix of the overall game



in very large games, submatrices will likely be non-overlapping

2. Each player solves its subgame (as if it were the whole game) and computes
   - mixed security levels: $V(A_1)$ & $V(A_2)$
   - corresponding security policies: y^*, z^*
3. Players play their mixed security policies against each other



Because of independent subsampling, a player can now be unpleasantly surprised:

$$\underbrace{E_{y^*,z^*}[a_{ij}] > V(A_1)}_{} + \epsilon$$

outcome larger than minimizer expected based on its submatrix A_1 (by more than ϵ)

SSP Notions of Security

Probabilistic notion of security:
- probability of (unpleasant) surprises should be below a pre-specified bound
- with more computational power, one can demand lower prob. of surprise

Definition: The SSP algorithm is ϵ – *secure for P1 (minimizer) with confidence 1–δ* if

$$\mathrm{P}(\underbrace{\mathrm{E}_{y^*,z^*}[a_{ij}] > V(A_1)}_{} + \epsilon) \leq \delta$$

outcome larger than P1 expected (by more than ϵ)

Definition: The SSP algorithm is ϵ – *secure for P2 (maximizer) with confidence 1–δ* if

$$\mathrm{P}(\underbrace{\mathrm{E}_{y^*,z^*}[a_{ij}] < V(A_2)}_{} - \epsilon) \leq \delta$$

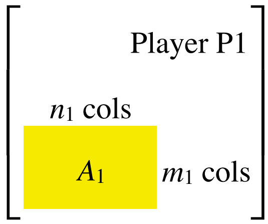
outcome smaller than P2 expected (by more than ϵ)

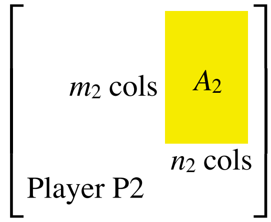
Can we guarantee ϵ – security for a pre-specified small probability of violation δ ?

YES, provided that our sample is sufficiently rich!

Game-independent Bounds

UC SANTA BARBARA engineering





Theorem: The SSP algorithm is $\epsilon = 0$ − *secure for P1 (minimizer) with confidence* $1-\delta$, for

$$\delta = \frac{m_1 n_2}{n_1}$$

Conversely, to obtain desired confidence level δ, suffices to select

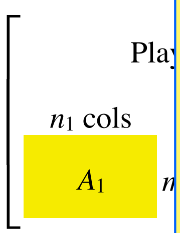
$$\frac{n_1}{m_1} \geq \frac{n_2}{\delta}$$

"fat" sampling for A_1
$\Downarrow$
test more options for opponent than own (by appropriate ratio)

Proof utilizes results from the "scenario approach" to convex optimization using randomized methods [Calafiori, Campi 2006-2009]

Game-independent Bounds

UC SANTA BARBARA engineering



Game-independent bounds

- valid for any game
- independent of the size of the game

Bounds on relative computation

- required size of my sample depends on size of opponents' sample
- the more I search for a good solution (large m_1), the more options need to consider for opponent (large n_1)

Theorem: The SSP alg... *with confidence* $1-\delta$, for

$$\delta = \frac{m_1 n_2}{n_1}$$

Conversely, to obtain desired confidence level δ, suffices to select

$$\frac{n_1}{m_1} \geq \frac{n_2}{\delta}$$

"fat" sampling for A_1
$\Downarrow$
test more options for opponent than own
(by appropriate ration)

Proof utilizes results from the "scenario approach" to convex optimization using randomized methods [Calafiori, Campi 2006-2009]

Case Study I: Combinatorial Search

UC SANTA BARBARA engineering

- P1 hides a treasure in one of M possible locations in the plane
- P2 wants to find treasure in minimum time (chooses among $M!$ possible paths)
- Classical example of *protecting high-value information*

For
- $M = 10$ possible treasure locations ($M! = 3.6$ million paths)
- 99% confidence ($\delta = 0.01$)
- P1 and P2 considers all possible treasure locations ($n_2 = m_1 = 10$)

P1 should sample

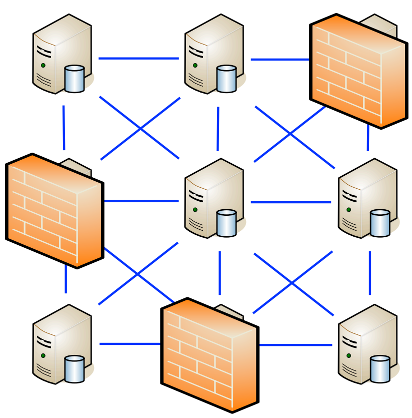
$$n_1 \geq \frac{n_2 m_1}{\delta} = 10000 \text{ paths}$$

to determine optimal hiding location

However, a posteriori bounds can provide good guarantees with much fewer samples

Case Study II: Dynamic Partial Inf. Game

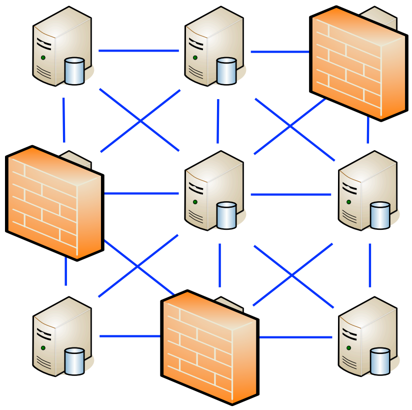
UC SANTA BARBARA engineering



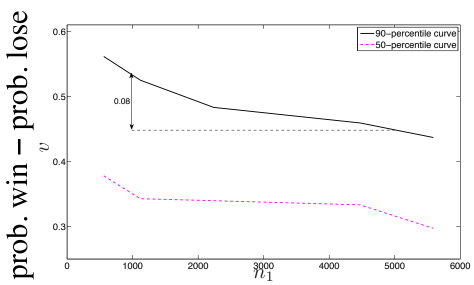
- Both players attempt to execute a mission, the one that completes it first wins
- Mission:
  - network with N mission-relevant computers
  - players takeover computers in turns
  - mission requires n computers to jointly execute a program, but only a few subsets of n computers can succeed
- Partial information:
  - in “open” computers, both players can see if other took over
  - in “closed” computers, a player cannot see if other already took over

think N=9, n=3, TTT

Case Study II: Dynamic Partial Inf. Game



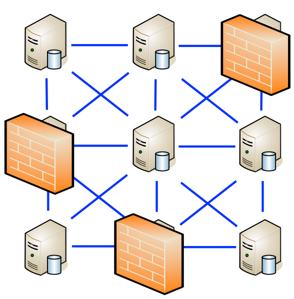
- Mission:
  - network with N mission-relevant computers
  - players take over computers in turns
  - mission requires n computers to jointly execute a program, but only a few subsets of n computers can succeed
- Partial information:
  - in “open” computers, both players can see if other took over
  - in “closed” computers, a player cannot see if other already took over



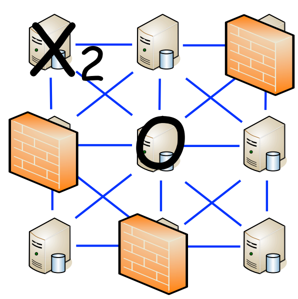
- Not possible to guarantee victory (for either player)
- First player has an advantage (≥ 50%wins)
- Optimal strategy involves randomized choices
- We have used SSP algorithm to construct players with 1% confidence

The Value of a Decision Aid System

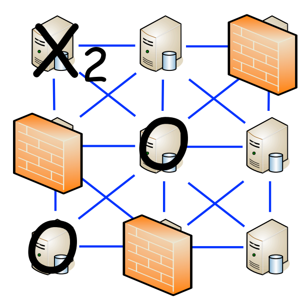
Stage 1: 1 X hidden



Stage 2: 1 X hidden



Stage 3: 2 Xs hidden

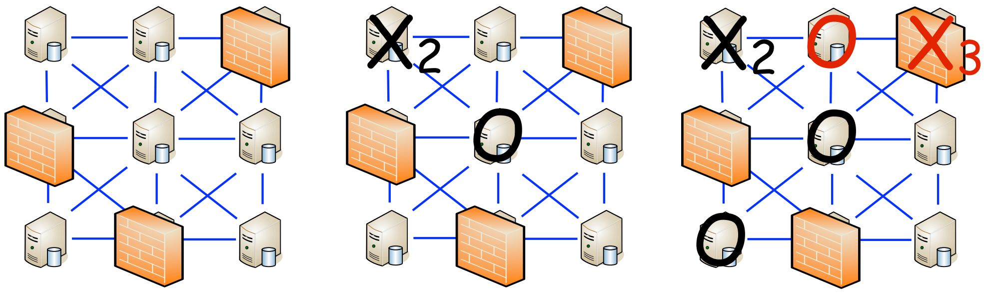


Q: What should O do?

The Value of a Decision Aid System

UC SANTA BARBARA engineering

Stage 1: 1 X hidden | Stage 2: 1 X hidden | Stage 3: 2 Xs hidden



In N = 9 computer network it is not too bad to keep track of these, but it is much harder for large N.

If there was a top-left X other player would not have hidden 2nd X

Q: What should O do?

A: No point in top left, X is there already.

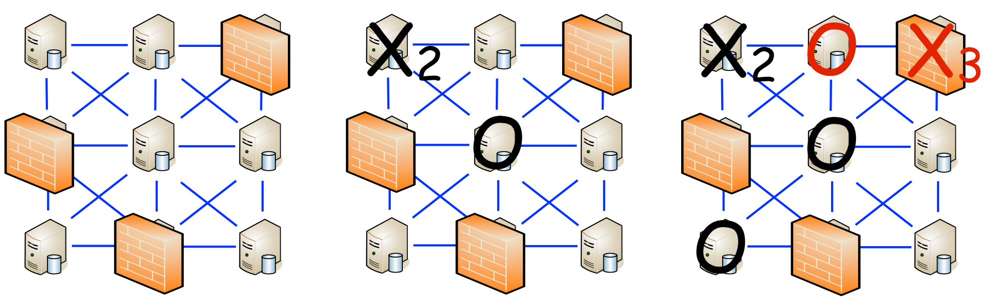
Must play top middle

No longer possible to win, draw is best bet.

The Value of a Decision Aid System

UC SANTA BARBARA engineering

Stage 1: 1 X hidden | Stage 2: 1 X hidden | Stage 3: 2 Xs hidden



In N = 9

Q: What should O do?

Project goal

- provide estimates of probability of mission success
- *help human operator sift through possible scenarios*

but

- no known solutions based on **separation** between estimation & control
  – must first find optimal players and then deduce optimal estimates

large N.

No longer possible to win, draw is best bet.

Conclusion

UC SANTA BARBARA engineering

- Cyber security as two-player large-scale matrix games
- SSP randomized algorithm provides probabilistic security guarantees
- Game independent theoretical bounds
  (in terms of relative computation of two players

Future work

- Tools for machine-aided decision (estimates of state, future actions, graphical interfaces, etc.)
- Mid-game players
- Uncertainty in other player's knowledge

Technical details on SSP at http://www.ece.ucsb.edu/~hespanha/published/#10GameTheory